CHAPTER 4

BY

VISHU GOYAL

ETISHA JAIN

LINUX 4.1 Understanding Common Text Tools

- **more** was the original file pager
- **less** was developed to offer some more advanced features
- As a reaction to that, more was developed a bit more.
- But still, to do more, use less.

- **less /etc/passwd**
- **press** page up and down arrow key
- press q for exit
- press space bar

- **head** to show the first 10 lines of a text file
- **tail** to show the 10 last lines
- use **-n nn** to specify another number of lines
- eg: **head -n 5 /etc/passwd**
- eg: **head -n 5 /etc/passwd | tail -n 1**
- eg: **tail -n 3 /etc/passwd**
- eg: **tail -f /var/log/messages** to show the messages in realtime.

- **cat** dumps text file contents on screen.
- **-A** shows all non-printable characters.
- **-b** numbers lines
- **-s** suppresses repeated empty lines
- **tac** is doing the same, but in reversed order.

- **cut :** filter output
- eg: **cut -f 3 -d : /etc/passwd | less**
- **sort :** sort the output
- eg: **cut -f 3 -d : /etc/passwd | sort | less**
- **tr :** translates
- eg : **cut -f 1 -d : /etc/passwd | sort | tr [a-z] [A-Z] | less**
- eg : **cut -f 1 -d : /etc/passwd | sort | tr [:lower:] [:upper:] | less**

LINUX 4.2 Using grep

- grep is excellent to find text in files or in output.
- ps aux | grep ssh it will filter out the ssh containing lines
- grep linda* it will find all the files which was started with linda...
- grep -i linda* it will find all the files and also a case insensitive.
- grep -A5 linda /etc/passwd for after
- grep -B5 linda /etc/passwd for before
- grep -R root /etc for recursively.

- grep linda *
- grep linda * 2>/dev/null -> it shows all the files and the lines contain the linda
- grep -l linda * to show only files listed
- grep -i linda * 2>/dev/null use for case insensitive.
- greap -Rl root /etc 2>/dev/null | less

LINUX 4.3 Understanding Regular expression

GREP stands for **generic regular expression parser**

- Regular Expression are txt patterns that are used by tools like grep and others
- Dont confuse regular expressions with globbing!

- Grep stands for generic regular expression parser
- Regular expressions apply to search patterns to something you're looking for inside a file while globbing just applies to the file names
- grep 'a*' a*
- Extremely good to put entire regular expression search pattern in single quotes .
- Single quotes make sure that the shell is not going to interpret the regular expression because if the bash shell reach a line like grep single quote or a star single quote then the shell is going to interpret all special characters like star and if the special character is between single quotes, the shell knows that it shouldn't text them which leaves them to be interpreted by regular expression treating utility.

- Regular expressions are for use with specific tools only: grep,vim,awk,sed
- Extended regular expressions enhance basic regex features. We need to specify a specific option to work with it like -e with grep utility.
- man 7 regex

- Regular Expressions are built around atoms . An atom specifies which text is to be matched.
- Atoms can be single characters,a range of characters, or a dot
- Atoms can also be a class , such as [[:alpha:]], [[:upper:]], [[:digit:]],[[:alnum:]]

- A repetition operator can be used which specifies how often a character occurs.
- Third element is indicating where to find next character.
- grep b.t regtext
- grep 'b.*t' regtext (* for 0 or more times)
- grep 'bo*t' regtext
- grep 'b.?t' regtext
- No output because ? is the part of extended regular expressions
- grep -e 'b.?t' regtext
- egrep 'b.?t' regtext (? for 0 or 1 occurrence)

Common Regular Expression

- ^ begining of the line
- $ end of the line
- \< begining of word
- \> end of the word
- * zero or more times
- + one or more times
- ? zero or one time
- {n} exactly n times

LINUX 4.4 Using awk

- Powerful text processing utility that is specialized in data extraction and reporting.
- it can perform actions based on selectors

- **awk -F : '/linda/ { print $4 }' /etc/passwd** --> print the 4th field of the file
- **awk -F : '{ print $NF }' /etc/passwd** --> **-F** stands for field seperator and **$NF** stands for no. of fields.
- (F for field separator and NF for number of fields)
- **ls -l /etc | awk '/pass/ { print }' | less**
- **ls -l /etc | grep pass**
- above both command are similar in nature.

LINUX 4.5 Using sed

- It come from Unix I.e. sed the stream editor
- Helps to apply modifications to text files.
- Sed is old version of vi and helps us to search and transform the text.
- it can be used to search for text , and perform an operation on matching text.
- **sed -n 4p sedfile** (4th line of file)
- **sed -i s/four/FOUR/g sedfile** option -i allows to write the text in the file.
- **-i** allows sed to write directly to the file.
- **sed -i -e '2d' sedfile**
- (-e for edit command to sed and it is 2d in this case)
- **sed -n 4p sedfile**

Thank you for refer these notes